Neural Networks & Deep Learning

ICP-2

Sreeja Reddy Konda

700756597

GitHub Link: https://github.com/SreejaReddyKonda/Neural-Network-Sreeja/blob/main/Neural%20Networks/ICP-2/ICP-2.ipynb

1.

```python
class Employee:
    # Count the number of employees
    employee_count = 0

    def __init__(self, name, family, salary, department):
        # Instance variables
        self.name = name
        self.family = family
        self.salary = salary
        self.department = department

        # When a new employee is created
        Employee.employee_count += 1

    def average_salary(self, *salaries):
        # Calculate and return the average salary
        total_salary = sum(salaries) + self.salary
        num_employees = len(salaries) + 1
        return total_salary / num_employees

class FullTimeEmployee(Employee):
    # Additional properties for FullTimeEmployee

    def __init__(self, name, family, salary, department, fulltime_property):
        # Call the constructor of the base class (Employee)
        super().__init__(name, family, salary, department)

        # Additional property specific to FullTimeEmployee
        self.fulltime_property = fulltime_property

    def average_salary(self, *salaries):
        # Calculate and return the average salary
        total_salary = sum(salaries) + self.salary
```

```python
    def average_salary(self, *salaries):
        # Calculate and return the average salary
        total_salary = sum(salaries) + self.salary
        num_employees = len(salaries) + 1
        return total_salary / num_employees


# Create instances of Employee class
employee1 = Employee("Sreeja", "Konda", 70000, "IT")
employee2 = Employee("Julie", "Smith", 80000, "HR")


# Call the average_salary function for Employee class
average_salary_employee = employee1.average_salary(employee2.salary)
print(f"Average Salary for Employees: {average_salary_employee}")

# Create instances of FullTimeEmployee class
fulltime_employee = FullTimeEmployee("Bob", "Joe", 80000, "Finance", "FullTimeProperty")

# Call the average_salary function for FullTimeEmployee class
average_salary_fulltime_employee = fulltime_employee.average_salary()
print(f"Average Salary for FullTimeEmployee: {average_salary_fulltime_employee}")

# Print the total count of employees
print(f"Total Employees: {Employee.employee_count}")
```

Output:

```
Average Salary for Employees: 75000.0
Average Salary for FullTimeEmployee: 80000.0
Total Employees: 3
```

Explanation:

This snippet is to give the average salary of the employees. So the code includes initialization of employees data while creating instances of employee and full time employee and thyen giving the output of employee count and average salary.

2.

```
[ ]  import numpy as np

     # Create random vector of size 20 with floats in the range 1-20
     random_vector = np.random.uniform(1, 20, 20)

     # Reshape the array to 4 by 5
     reshaped_array = random_vector.reshape(4, 5)

     # Replace the max in each row by 0
     reshaped_array[reshaped_array == reshaped_array.max(axis=1, keepdims=True)] = 0

     reshaped_array
```

Output:

```
array([[ 3.74773321,  4.30459136,  3.95087555, 16.44491375,  0.        ],
       [ 3.85969346,  8.20696962,  0.        ,  2.79792357,  6.93204385],
       [15.15034058,  2.3918844 , 17.41152261,  0.        , 15.73649723],
       [13.11207156,  0.        ,  7.71750655,  3.03076777,  1.06445067]])
```

Explanation:

The above snippet is to give the output of an (4,5) array where the maximum values will give 0 as output.